

# Neural Network Applied to the Coevolution of the Memetic Algorithm for Solving the Makespan Minimization Problem in Parallel Machine Scheduling

Tatiane R. Bonfim & Akebo Yamakami  
School of Electrical and Computer Engineering  
State University of Campinas - Campinas - SP, Brazil  
tatiane, akebo@dt.fee.unicamp.br

A scheduling problem treats the allocation of resources, taking into account the execution of a set of operations, and with an objective, subject to a set of restrictions. The problem discussed here is one of scheduling the tasks in identical parallel machines. In this problem, we worked with a set of  $n$  tasks and  $m$  identical parallel machines, with the objective of minimizing the makespan. The makespan is the total processing time of the most busy machine. We considered that the tasks should be assigned without preemption, each machine can process one or more tasks, one at each time, and each task should be processed by just one machine.

This work presents an implementation of a memetic-neuro scheduler for solving this scheduling problem. The memetic algorithm has been used to evolve good scheduling forms and the neural network has been used to calculate the fitness for each individual of the population.

The memetic algorithm is an hybrid version of genetic algorithm with local search with the objective to increase the performance of the genetic algorithm. In the hybrid method, the genetic algorithm is used to execute the global exploration in the population, while the heuristic method is used to execute the local exploration in the chromosomes. For the local search, we selected an improvement heuristic that uses the first improving type strategy.

One of the objectives of this work is to verify that using neural network co-evolving with the memetic algorithm, it is possible to estimate the fitness function of the problem, which is not always known.

The adopted memetic scheduler architecture possesses an input layer with three neurons. The first one represents the total processing time of the tasks, the second one represents the mean time and the third represents the total processing time of the less busy machine. The hidden layer possesses 20 neurons and the output layer has just a neuron that represents the individual's fitness in the memetic algorithm.

The neural network acts in the co-evolution of the memetic algorithm. In the learning phase, the neural network uses the entrances obtained through scheduling accomplished by the memetic algorithm, either for an

individual of the initial population or for the sub-population, to learn the fitness value associated to those entrances.

Two neural networks were applied with that purpose. The first one was the back-propagation, that uses an expected value as output to skill the network and the second one was with reinforcement learning, based on the interaction between agent and environment.

In the beginning of the back-propagation algorithm, we used the final processing time of the last executed task as expected output value.

In the reinforcement learning, to define the environment we created two rules to compare the output of the neural network with the expected output. The first rule assumes acceptance rate if the output of the neural network was smaller or equal that the expected output and, otherwise, assumes rejection rate. It was used as waited output, the value of the makespan calculated with the scheduling accomplished by the memetic algorithm. During the execution of the scheduler, according with the state of the net, an action is returned (acceptance or rejection). These returned rates have been considered as the rewards given to the agent.

We performed 390 instances of test, where minimum makespan of each one is known. The number of processors, in these instances, was 5, 10 or 25. The number of tasks was 10, 50, 100, 500 or 1000. The processing time of the tasks were randomly generated in agreement with an uniform distribution in the intervals,  $[1, 100]$ ,  $[1, 1000]$  and  $[1, 10000]$ .

In this work we showed that neural network works very well in the fitness function determination. Simulations showed that the neural network presents a good performance in the co-evolution of the memetic algorithm, if compared with the results obtained with memetic algorithm without neural net. The simulation results showed that both neural nets implemented learned well and found good fitness solutions. The neural net with reinforcement learning worked a little better than the backpropagation net.